

Quasar

Juni/Juli 1988 nr.36



My men all use
MS DOS,
or they don't use
anything at all.

INHOUD QUASAR 36

Kolofon	738
Van de Redactie	739
Nieuws van de Microfair	740
Ombouw naar JS versie	741
Gebruik van PSION's	
"Printer_dat" in BASIC	742
Hardware uirbreidingen	743
MG ROM	744
Ervaringen van een	
deblokkeur	746
CHAPRO	748
Ombouw CST Disc-	
Controller	750
Oproep	751
Procedure om een matrix	
te inverteren/	
vermenigvuldigen	752
CURSUS MACHINETAAL	
Hoofdstuk 9 subroutines	758
Vraag en Aanbod	760

SLUITINGDATA KOPIJ

6 augustus
3 september
8 oktober

KOLOFON

Stichting SIN__QL__AIR
Rotterdam
giro: 4597345

ADMINISTRATIE
sekretariaat
PENNINGMEESTER
Nabestellen oude
nummers.

BOB VISSER
Snelrewaard 6
2904 SN Capelle a/d
IJssel
Tel. 010-458.3161

VOORZITTER

RON DEN BREEMS
Kroonstaddreef 27
3067 RT Rotterdam
Tel. 010 - 455.1234

REDAKTIE, layout en
samenstelling Quasar.

GERARD VAN ROOIJEN

Gruttostraat 15
3435 DJ Nieuwegein
Tel. 03402 - 33027

DATABANK

Tel. 03404 - 22533
Sysop:
MICHEL & WILLEM
SPANJER
Hortensialaan 11
3702 VD Zeist
Tel. 03404-20581
SVP aléén tussen
19.00 en 22.00 uur

HARDWARE
Reparaties en
onderdelen

MICHEL & WILLEM SPANJER

Hortensialaan 11
3702 VD Zeist
Tel. 03404 - 20581
SVP alléén tussen
19.00 en 22.00 uur

VRAGEN OVER:

Superbasic, Pascal,
Machinetaal, Quill,
Archive, Abacus,
Hardware

KEES V.D. WAL

Kwekerijstraat 22
2613 VE Delft
Tel. 015 - 140367
SVP alléén tussen
20.30 en 23.30 uur

VRAGEN OVER:

Machinetaal en Hard-
ware

ARD JONKER

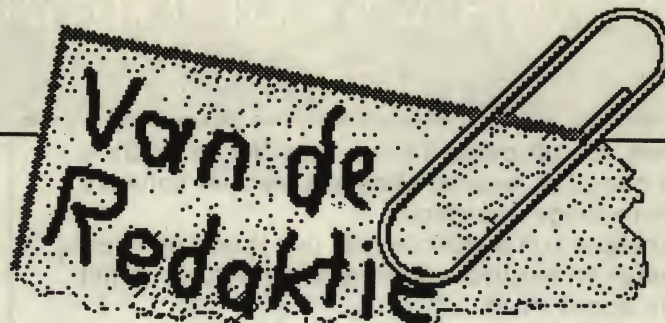
Tel. 020 - 230795

VRAGEN OVER:

Machinetaal en C

MARC KOOL

Tel. 020 - 429345



Zo de nieuwe uitgave van de Quasar ligt weer voor U. Het was op 4 juni gezellig druk op de bijeenkomst in Utrecht, helaas was de tijd tussen het verschijnen van de vorige Quasar en de bijeenkomst erg kort, voortaan zullen wij trachten hier wat meer tijd tussen te laten zitten.

Op de bijeenkomst zijn er diverse vragen op ons afgevuurd, met name door beginnende QL-ers, deze gebruikers zijn zeer op zoek naar informatie en de vraag leeft of er wat meer artikelen speciaal afgestemd op deze groep gepubliceerd kunnen worden. Nu zal een ieder zich zijn eerste ervaringen (en evt. blunders) nog wel herrinneren zo dat het ons een goed idee lijkt dit eens op papier (of cartridge eigenlijk) te zetten en aan ons toe te zenden zodat wij er van mee kunnen genieten of in ieder geval iets van kunnen leren.

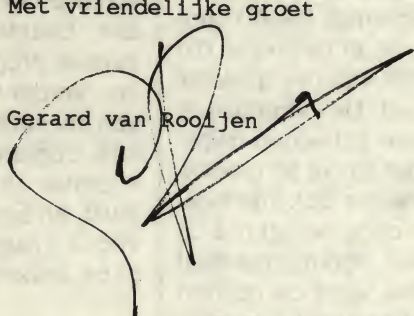
Wij zijn op het ogenblik Succes van Digital Precision aan het testen, het verhaal van deze CPM Code generator kunt u in het volgend nummer van de Quasar verwachten.

Alle andere artikelen over het QL-gebeuren blijven natuurlijk van harte welkom.

Uw inzendingen gaarne als Quill-doc of als ASCII-file op cartridge of 5 1/4 inch floppy. Ook een uitgeprinte versie bijleveren is voor ons erg handig.

Met vriendelijke groet

Gerard van Rooijen



NIEUWS VAN DE MICROFAIR

De microfair, een sinclair beurs die voornamelijk op de QL is gericht, in een oude hal, waarbij bijna alle hard en software producenten zijn vertegenwoordigd.

Als sinclair liefhebber kun je daar de hele dag je hart ophalen, met programmeurs en hardware ontwerpers praten, die je net zo vriendelijk te woord staan...

Wat was er te zien deze keer:

Een ATARI ST die draaide op Qdos. Het zag er degelijk en vooral erg echt (en dus eng) uit. Zelfs de kleuren zagen er origineel uit, Qram draaide er op, en dat is toch een programma dat Qdos behoorlijk ondersteboven haald en behoorlijk huishoud in het geheugen, en multitasking draaid etc. etc. Het enige voor mij zichtbare was dat sommige toetsen incompatible waren, wat bijvoorbeeld voor de THOR zonder alt-key toch een klein nadeel was m.n. compatibiliteit. Voordelen zijn: het beschikbaar zijn van een harddisk, 4 Megabyte geheugen, en als hardcopy is er voor de ATARI ook een Laserprinter verkrijgbaar. Voor de ATARI gebruiker is het voordeel van QDOS het eindelijk te kunnen multitasken (wat al erg lang voor de ATARI is beloofd) en een sterke basic, tevens heb ik van horen zeggen dat de desktop publishing voor de QL veel verder is gevorderd dan die van de ATARI...

Het nieuws op de SANDY stand was echter veel opzienbarender met 2 nieuwe interfaces opstapel staand:

1) Een PC XT Interface, waarmee men op de PC, Qdos kan draaien met ms-dos als eenmalige job. Het geheel bevindt zich op n print, waarop zich alle aansluitingen bevinden, een grotere broer van de 68008, plaats voor een extra rom, en doordat de print compleet volgepakt is met ic's worden de printbanen op meerdere lagen van en naar de ic's geleid. Vol-

gens de ontwerper die ik persoonlijk mocht aanhoren ging het om een 6-layer print. De kaart neemt eenmaal ingeschakeld de 'macht' van de PC over en kan gebruik maken van de standaard randapparatuur zoals drives, winchesters, monitor etc. etc. De goedkope klonen die op dit moment verkrijgbaar zijn voor onder de f2000,- met 20 Mb harde schijf en alle extras als muis e.d. erbij, maken de kaart interessanter dan dan een THOR - 20 met ms-dos (zoals nu c/pm voor de QL) als emulatie pakket. Voor de PC eigenaren betekend het dat men toch kan multitasken (zoals men ons met OS/2 beloofd) op de oude PC, en de oude programma's kan behouden.

Het prototype was op een enkel schoonheidsfoutje af, en werd op de stand in geuren en kleuren van commentaar voorzien.

2) Het SANDY MEGA board met Harddisk interface. Deze kaart is SANDY's antwoord op de trumpcard van MIRACLE met meer geheugen, de mogelijkheid om extra expansions bij te steken (rom's, eprom programmers, DA/AD convertors etc.), en al de extras die we gewend zijn van het 'gewone' Super Q Board. De interface is door de nieuwe geheugen chips, zo compact dat hij net zo groot is als een ouderwetse disk interface! De geheugen chips brengt ons op de datum van verkrijgbaarheid van de kaarten, want de kaarten zelf worden al gemaakt, voorlopig wordt gekocht op de eerstvolgende microfair begin herfst. Men kan

dan kiezen tussen een MEGA BOARD voor tegen de duizend piek, of met Harddisk 20, 30 of 40 Mb voor ongeveer tweeduizend piek, waarbij voor iedere 10 Mb meer ongeveer 40 pond extra moet worden neergelegd. De prijzen erg globaal en hangen nog van de geheugen IC's en de koers af. Het geheugen kan worden gedeeld in bijvoorbeeld 640k plus 512k ramdisk of in z'n geheel door ingebouwde software, de processor te laten 'denken' meer geheugen voor handen te hebben. Alle software voor de QL is net zo compatible als anders, op deze nieuwe kaart. Mijn hart is in ieder geval verloren, aan het laatste board.

S.U.B. eindelijk ben ik ook er achter gekomen waar al die groote advertenties in de QL world voor dienden. Het zijn 2 mensen die full time in de postorders zitten en tussen door QL'ers met problemen te woord staan, en vervolgens iedere maand een blaadje van 40 pagina's A5 uitbrengen. Het blad zit tussen de commerciële QL world en onze vertrouwde QUASAR in, en bestaat volledig uit vaste rubrieken.

Volgende maand kunnen de basic extensions voor flashback worden verwacht, waarmee men verschillende kaartenbakken kan gaan koppelen en zelfs kaarten bakken aan bestaande of nieuw gemaakte programma's kan koppelen. Verder was er weinig nieuws van het software front, op een enkele upgrade na. De volgende microfair is voor mij weer een must, en om het maar eens recht voor z'n raap te zeggen: be there or be square!

Paul Moerman,
tel: 010-4348803

Om bouw naar JS versie.

GEBRUIK VAN PSION'S "PRINTER_DAT" IN SUPERBASIC

Bij het maken van een administratief programma voor mijn broer, kwam ik voor het probleem te staan, dat er voor het uitprinten van de gewenste lijsten (waarin een gedeelte onderstreept, vergroot en of vet gedrukt moest worden) 2 reeksen printer-control codes nodig waren. Een reeks voor de printer van mijn broer en de andere voor mijn printer.

Ik had dus voor elke printer een aparte procedure gemaakt. Eigenlijk had ik dus voor elke printer een ander programma. Dit werkte wel maar was wel omslachtig. Toen kwam ik op het idee waar het PSION-pakket van de QL ook mee werkt.

Met behulp van 'install_bas' moet men een eigen printer-code lijst aanmaken, die vervolgens wordt gebruikt wanneer men uit wil printen. Na een paar uur zoeken en

vergelijken in het programma: install_bas en de file: printer_dat, wist ik hoe deze laatste file is opgebouwd.

In het onderstaande programma wordt nu deze file gesplitst en de printer-code's in (alfa-)nummerieke variabelen gezet die al dan niet in PRINT statements kunnen worden gebruikt of kunnen worden aangeroepen.

Het is wel van "belang dat wanneer bv. "erline off" niet gebruikt

worden, de procedure "join" die in diezelfde regel wordt aangeroepen te laten staan. Anders worden de resterende printer-control code's in de verkeerde variabelen gezet worden met alle gevolgen van dien.

Het programma is, dacht ik, voldoende gedocumenteerd zodat het voor de hobbyist mogelijk is het programma te schrijven in andere talen zodat de programma's professioneler of niet makkelijker toegankelijker voor andere printergebruikers.

Veel succes

Dick Nieuwensteeg

```

100 REMark INITIALISATIE VAN VARIABLEN
      VOOR HET GEBRUIK VAN
110 REMark PSION'S PRINTER_DAT IN
      SUPERBASIC
120 REMark DICK NIEUWESTEEG
130 REMark JUNI 1988
135 :
140 PRINTER_INIT
150 PRINT #3;"PRINTER: ";NAME$;
      " GEINITIALISEERD"
160 PRINT #3;UNDERON$;BOLDON$;"DEZE TEKST
      IS ONDERSTREEPT EN IN BOLD
      GESCHREVEN";BOLDOFF$;UNDEROFF$
170 REMark EEN VOORBEELD HOE EEN FUNCTIE
      VAN TRANSLATE KAN WERKEN IS:
180 REMark IF TEXT(X TO X)=TRANSLATE1$(1)
      THEN PRINT #3;TRANSLATE1$(2 TO)
190 :
200 DEFine PROCEDURE INIT_PRINTER
210   OPEN #4,FLP1_PRINTER_DAT
220   LET PRINTER$=""
230   REPEAT LOOP1
240     IF EOF(#4) THEN EXIT LOOP1
250     PRINTER$=PRINTER$&INKEY$(#4)
260   END REPEAT LOOP1
270   CLOSE #4
280   NAME$=PRINTER$(6 TO 15)
      :REMark PRINTER NAME

```

```

290   PORT=CODE(PRINTER$(16))
      :REMark PORT=1 : SER1
300   :REMark PORT=2 : SER2
310   :REMark PORT=0 : PARALLEL OR NON
      STANDARD SERIAL INTERFACE
320   IF PORT=1 OR PORT=2
330     PARITY=CODE(PRINTER$(17))
      :REMark
      0=NONE, 1=SPACE, 2=MARK, 3=ODD, 4=EVEN
340     BAUDRATE=CODE(PRINTER$
      (18))*256+CODE(P_ BAUD RATE
350     BAUD BAUDRATE
360     OPEN #3,"SER"&PORT
370   ELSE
380     DEV$=PRINTER$(18 TO 18+CODE
      (PRINTER$(17)))
      :REMark NAME OF PARALLEL OR
      NON-STANDARD SERIAL INTERFACE
390     OPEN #3,DEV$
400   END IF
410   LPP=CODE(PRINTER$(35))
      :REMark LINES / PAGE
420   CPL=CODE(PRINTER$(36))
      :REMark CHARACTERS / LINE
430   CONFIRM=CODE(PRINTER$(37))
      :REMark CONTINUOUS FORMS 1=YES, 2=NO
440   POS=42
450   JOIN:EOL$=LINES
      :REMark END OF LINE CODE

```



```

460 JOIN:PRECODE$=LINE$
:REMark PREAMBLE CODE
470 JOIN:POSTAMBLE$=LINE$
:REMark POSTAMBLE CODE
480 JOIN:BOLDON$=LINE$
:REMark BOLD ON
490 JOIN:BOLDOFF$=LINE$
:REMark BOLD OFF
500 JOIN:UNDERON$=LINE$
:REMark UNDERLINE ON
510 JOIN:UNDEROFF$=LINE$
:REMark UNDERLINE OFF
520 JOIN:SUBSCRIPTON$=LINE$
:REMark SUBSCRIPT ON
530 JOIN:SUBSCRIPTOFF$=LINE$
:REMark SUBSCRIPT OFF
540 JOIN:SUPERSCRIPTON$=LINE$
:REMark SUPERSCRIPT ON
550 JOIN:SUPERSCRIPTOFF$=LINE$
:REMark SUPERSCRIPT OFF
560 JOIN:TRANSLATE1$=LINE$
:REMark TRANSLATE1
570 JOIN:TRANSLATE2$=LINE$
:REMark TRANSLATE2
580 JOIN:TRANSLATE3$=LINE$
:REMark TRANSLATE3
590 JOIN:TRANSLATE4$=LINE$
:REMark TRANSLATE4
600 JOIN:TRANSLATE5$=LINE$
:REMark TRANSLATE5

```

```

610 JOIN:TRANSLATE6$=LINE$
:REMark TRANSLATE6
620 JOIN:TRANSLATE7$=LINE$
:REMark TRANSLATE7
630 JOIN:TRANSLATE8$=LINE$
:REMark TRANSLATE8
640 JOIN:TRANSLATE9$=LINE$
:REMark TRANSLATE9
650 JOIN:TRANSLATE10$=LINE$
:REMark TRANSLATE10
660 PRINTER$="":LINE$=""
670 END DEFINE

680 DEFINE PROCEDURE JOIN
690 REMark SAMENVOEGEN VAN PRINTER-CONTROL
CODE'S
700 LINE$=""
710 LENGTH=CODE (PRINTER$ (POS))
720 IF LENGTH>0 THEN
730 REMark GEEN PRINTER-CONTROL
GEDEFINIEERD
740 ELSE
750 FOR J=1 TO LENGTH
760 LINE$=LINE$&PRINTER (POS+J)
770 END FOR J
780 END IF
790 POS=POS+LENGTH+1
800 END DEFINE

```

Hardware uitbreidingen

De redactie ontving een schrijven van Jan Peter Venema uit Veendam waarin hij te kennen gaf dat hij wat hardware projecten op wil gaan starten en deze in de Quasar te publiceren, nu wilde hij graag weten waar specifiek belangstelling voor is zodat hij zijn ontwikkelingen voor een zo'n breed mogelijk publiek kan maken. Indien u belangstelling heeft voor een hardware ontwikkeling dan gaarne uw reactie naar het redactieadres of rechtstreeks naar Jan Peter Venema

Enige suggesties zouden kunnen zijn:

- seriële interface
- AD & DA converter
- MIDI
- 2e scherm voor QL
- keyboardinterface
- bufferprint
- digitizer
- enz.

- Parallele interface
- Timer
- Floppydisk interface
- Harddiskinterface
- EPROM programmer
- RAM uitbreiding
- scanner
- enz.

Redactie Quasar
Gruttostraat 15
3435 DJ Nieuwegein

Jan Peter Venema
Westereems 1
9642 KP Veendam

MG ROM

De MG Rom is de laatste Rom voor de QL die door Sinclair is uitgebracht. De meeste QL gebruikers werken met de JM of JSrom. De MG Rom is eigenlijk speciaal voor de Duitse markt aangepast voor hij op de Engelse markt zou gaan rouleren maar helaas van dit laatste is het nooit gekomen. Maar nu is het dan zover er is een Engelse MG Rom genaamd MGUK (MG United Kingdom).

De oorspronkelijke Duitse versie is aangepast door Simon Goodwin en is een Engelse versie geworden, die uiteraard ook door ons gebruikt kan worden.

De extra Basic extensions die hier in (t.o.v. JS) zijn:

Tron (Trace on)

Troff (Trace off)

Trace_speed (Snelheid van Tracen)
(0=snelst ;
32767=traagst)

Trace_dev (Verandert het kanaal
waarnaar de informatie
moet worden geprint)

Tdir (Geeft een directory in
3 kolommen)

(vb:Tdir flp1_)

file1	file2	file3
text1	text2	text3
etc.	etc.	etc.

Move_mem (Kopieert een stuk van
het geheugen naar een
ander)
(vb:Move_mem
131072,147456,16384)

Do_reset (Reset de computer
zonder de reset knop
te gebruiken, dus een
reset vanuit de basic)

Free_mem (Geeft aan hoeveel
vrije geheugen er nog
is)(vb:Print free_mem)

Rechp (Heeft iets met de com-
mon heap te maken
(read comm.heap ??))

Alchp (Heeft iets met de com-
mon heap te maken
(allocate comm.heap?))

Dev_stat (???? ben ik helaas
nog niet achter)

De MG Rom heeft ook een extra ka-
rakter set en wel de Griekse karakter
set, door de ALT en CAPS LOCK
toets in te drukken kan je tussen de 2
karakter sets wisselen.

Ook is het mogelijk cartridges te for-
materen met het aantal sectoren dat
je zelf wilt.

En nog een aantal extra's.

Het probleem is om de MG Rom code
(die reeds in mijn bezit is) in Rom te
krijgen, nu dit is niet mogelijk maar
het is wel mogelijk om de code in Ep-
rom te blowen.

Nog een probleem is dat een Eprom
niet rechttoe rechtaan in de plaats
van de Roms geplaatst kunnen word-
en.

Ook hier is een oplossing voor.
Ten eerste zijn er twee Rom een 16k
(9128C) en een 32k (9256C).

Dus hebben we ook twee Eproms no-
dig en wel een 16k (27128) en een
32k (27256).

De 32k Rom is de "als je de QL recht
voor je hebt met de expansiebus links
" linkse Rom en de 16k Rom zit
rechts van de 32k Rom.

We nemen eerst de 16k (27128) Ep-
rom.

Nodig is:

- De 16k Eprom met de code erin
- Een IC type 74LS00
- Een 28 pins IC voet

Soldeer op Eprom pin 14 een draadje
naar 74LS00 pin 7

Soldeer op Eprom pin 15 een draadje
naar 74LS00 pin 8

Soldeer op Eprom pin 21 een draadje
naar 74LS00 pin 11

Soldeer op Eprom pin 16 een draadje
naar Eprom pin 23

Soldeer op 74LS00 pin 9 een draadje
in IC voet pin 21

Soldeer op 74LS00 pin 10 en draadje
in IC voet pin 23

Buig van de Eprom pin 21 en 23 om
zodat deze niet in
de IC voet komen en stop dan de Ep-
rom in de IC voet.

Plaats dit geheel op de plek van de
16k (9128C) Rom.

(ZIE BIJGAAND SCHEMA)

Nu de 23k (27256) Eprom.

Nodig:

- De 32k Eprom met de code erin.
- Een IC type 74LS00
- Een 28 pins IC voet

Soldeer op Eprom pin 14 een draadje
naar 74LS00 pin 7

Soldeer op Eprom pin 15 een draadje
naar 74LS00 pin 8

Soldeer op 74LS00 pin 8 een draadje
naar 74LS00 pin 9

Soldeer op 74LS00 pin 10 en draadje
in IC voet pin 21

Buig van de Eprom pin 21 om zodat
deze niet in de IC
voet komt en stop dan de Eprom in
de IC voet.

Plaats dit geheel op de plek van de
32k (9256C) Rom.

(ZIE SCHEMA)

En klaar zijn we zonder te solderen aan de QL print.
Dit verhaal geldt voor alle issu's met uitzondering van de QL's waarin al Eproms in plaats van Roms zitten, deze kunnen direct de Eproms verwisselen.

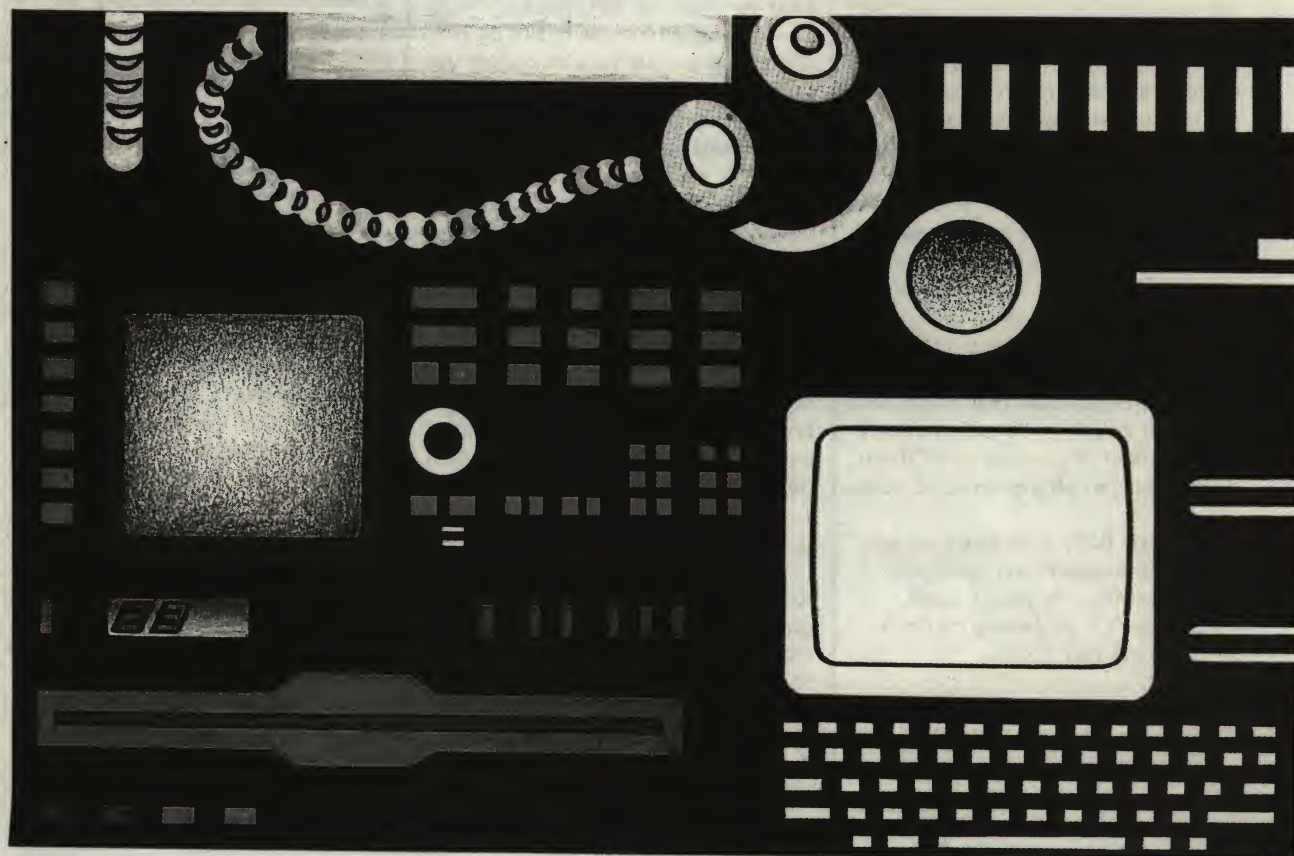
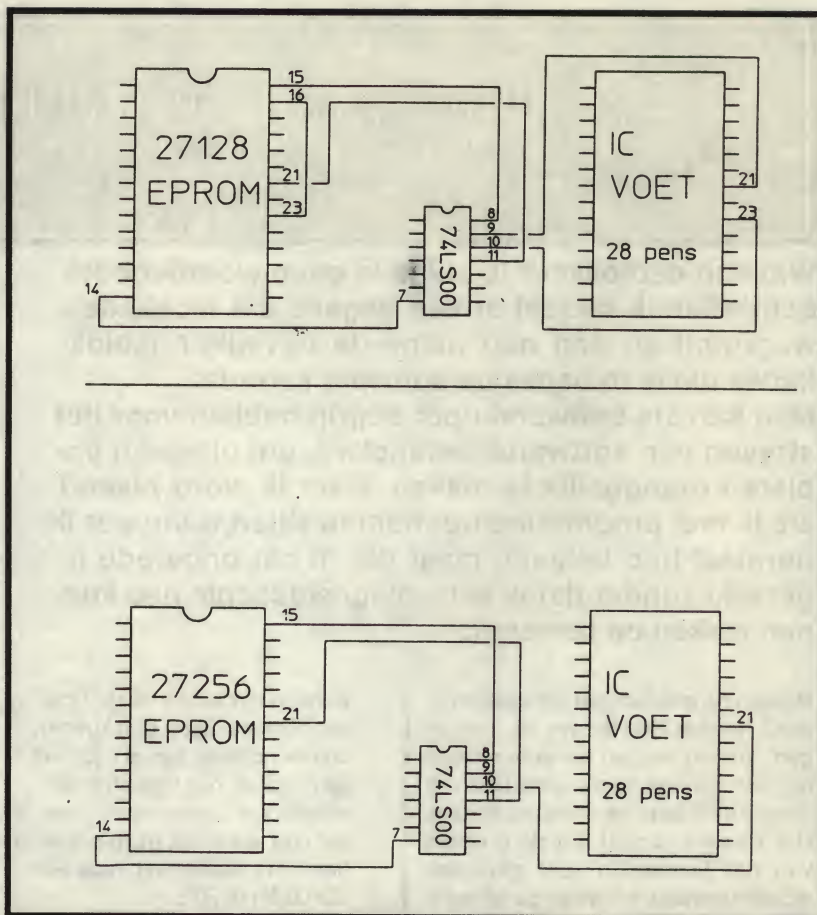
Wat gaat het ongeveer kosten:

- 2 Eproms F 40,00
- 2 IC voet 28p F 5,00
- 2 IC 74LS00 F 2,00

----- +
Totaal F 47,00

De MGUK Rom code is bij mij verkrijgbaar en kost niets op een retour postzegel na.(stuur wel een floppy of cartridge mee)

Fred van der Neut
Postbus 2072
2930 AB Krimpen aan de Lek
Tel. 01807-10553



Ervaringen van een deblokkeur

Wat een deblokkeur is zul je in geen woordenboek aantreffen. Ik bedoel ermee degene die blokkades wegruimt en dan met name de beveiligingsblokkades die je in bepaalde software aantreft.

Men kan als softwarekoper begrip hebben voor het streven van softwareleveranciers, om (illegaal) kopiëren onmogelijk te maken. Maar ik word razend als ik met programmatuur kom te zitten, waarvoor ik normaal heb betaald, maar die in het ongereede is geraakt zonder dat ik een veiligheidskopie heb kunnen maken en bewaren.

Moeten de goeden met de kwaden lijden? Neen, ik vind dat we ons met eigen kunnen mogen verweren tegen de niet geringe, soms artistieke vindingrijkheid van de softwaremakers. Het spreekt vanzelf dat de grenzen van het fatsoen in acht genomen moeten worden: wij willen de leveranciers niet benadelen, alleen tegen gaan dat zij ons benadelen.

Wel, zo zet je je dan vol goede moed aan de taak, om een groot en ingewikkeld programma in machinecode zodanig te ontrafelen dat blokkades kunnen worden gelocaliseerd en opgeheven. In mijn geval was het PSI-ON-Chess programma het eerste werkstuk dat ik onder handen nam. Ik dacht wel wat van m.c.-programmeren te weten, maar het karwei viel knap tegen. Want je duikt in een programma dat de makers niet voor vreemde ogen bestemd hebben en dat in verband daarmee nogal wat vrijheden bevat in de structuren. Ik kwam dingen tegen die ik mijn studenten vroeger streng verbood, zoals

- * JSR- en BSR- instructies, waarbij geen terugkeer naar de opvolgende instructie plaats heeft, doordat b.v. onderweg de stackpointer is veranderd.
- * Geneste JSR- en BSR- aanroepen, waardoor ook de RTS- instructies ondoorzichtig worden.
- * Niet-conditionele JMP's en BRA's; deze verstoren het over-

zicht op hinderlijke wijze. Ook conditionele sprongen kunnen onoverzichtelijk zijn, n.l. als het sprongdoel niet logisch in de structuur is opgenomen. Erger is het nog als wordt gesprongen, al dan niet conditioneel, naar een JSR, BSR of RTS.

Dit soort structuurzonden, die alle best vermijdbaar zijn, sluipen binnen in een programma als het in de ontwikkelingsfase herhaaldelijk wordt veranderd en verbeterd (wat de werking betreft). Ik zat daardoor voor een enorm bord spaghetti.

De zaak werd nog erger door een aantal op zich wel correcte, maar bij de ontrafeling niet meteen prettige details, zoals vertragslussen, het be- of overschrijven door het programma van delen van zichzelf en het feit dat flinke stukken Q-DOS gebruikt worden.

Wel is dat Q-DOS er uiteraard om gebruikt te worden, maar bij de exploratie kan je uiteraard in die gedeelten geen breekpunten plaatsen en het is erg lastig vast te stellen waar het programma Q-DOS weer verlaat.

Ken je eenmaal de weg in het programma, dan vervallen veel van deze moeilijkheden (ze telden niet voor de schrijvers), maar het kost tijd, veel tijd en geduld, om zover te komen. Afgezien nog van de praktische resultaten die wenken aan de horizon is er een beloning voor de zwoegeer: je leert ervan en elke volgende klus gaat beter.

Voor zo'n karwei is een goed monitorprogramma onontbeerlijk. Ik gebruikte MONQL v2.2 van HISOFT, en dit bleek een fijn hulpmiddel met m.i. maar twee (kleine) bezwaren. Het eerste is dat bij activeren van MONQL een inleidend scherm een volle 15 seconden blijft staan en dat is (te) lang wachten als je verder wilt. Het tweede bezwaar is dat terugkeer naar Basic niet meer lukt nadat het programma op een breakpoint is gestopt. Je moet dan resetten en opnieuw beginnen, wat frustrerend is. Als Andy Pennell me niet voor is, doe ik daar misschien nog wel eens iets aan (hoewel ik ook rekening houd met de mogelijkheid dat mijn 256K niet voldoende is om MONQL en CHESSC naast elkaar te laten functioneren).

De methodiek (althans voor mij) is niet strak te omschrijven: meer losse polswerk, improviserend. Ergens beginnen, stukje disassembly op het scherm en breakpoint plaatsen: wat doet hij? Conclusie trekken, nieuw breakpoint plaatsen: wat doet hij nu, waar komt hij langs? En dit maar herhalen, af en toe ondersteund met het uitprinten van een stuk disassembly. Soms na halt in breakpoint singlestepping verder.

Het zoeken wordt steeds gericht, het beeld duidelijker. Het blijft en blijft echter zeer complex verweven en op vele plaatsen slecht gestructureerd. Op een bepaald moment ga je dan ook verwachte instructies localiseren met de zoekfaciliteit van de monitor. En je ontdekt terloops, dat singlestepping niet altijd tot het goede pad leidt, doordat de timing ergens afwijkt.

Ik was met dit alles een eind op weg toen ik het verhaal las van Ed Vermeulen in Quasar 32. Het is natuurlijk fijn te merken dat een ander met hetzelfde bezig is, misschien hoeft ik niet verder, dus: meteen de programma'tjes van Ed geprobeerd. Helaas, het werkt (bij mij) niet. Dus maar verder ploeteren, berichtje in databank, rectificatie van Ed in Quasar, weer

proberen, geen resultaat.

Dus maar op eigen kracht verder, tot het zoete einde. Want uiteindelijk wist ik niet alleen waar de protectietests zaten, ik had ze ook uitgeschakeld, EN HET WERKTE. Terloops merkte ik nog even, dat met MONQL en CHESS beide in het geheugen het runnen van CHESS niet foutloos lukte: ze zitten elkaar in de weg. Het is dus ook niet zo vreemd te veronderstellen (zie een paar alinea's terug), dat MONQL wordt gehinderd door het vrij grote CHESSC.

Bij het voorafgaande ontbreekt nu nog de gedetailleerde informatie over de uitgevoerde programma-aanpassing. Ik heb niet gekozen voor publicatie van een programma in machinecode (al dan niet met hex-loader), dat de aanpassing aanbrengt, als het ware in den blinde. Het werk waarvan dit verhaal een uitvloeisel is, blijkt zo fascinerend, dat het zonde zou zijn het af te sluiten met een recept: daar leert niemand wat van. Tegen Ed Vermeulen zeg ik in dit verband, dat hij mij geholpen heeft met de aanduiding omtrent de aard van de uit te schakelen controles; ik heb echter helaas niet kunnen ontdekken, hoe zijn recept het gewenste resultaat zou kunnen bewerkstelligen.

Het uitschakelen van controles kan in beginsel zowel gebeuren door er overheen te springen (soms met JMP of BRA, soms met vervanging van bestaande instructies door 4E71, zijnde NOP), als ook door ervoor te zorgen dat de test altijd goed uitvalt. Laatstgenoemde aanpak heeft het voordeel ook bruikbaar te zijn in minder overzichtelijke situaties: men loopt minder kans per ongeluk ook nodige nevenwerkingen uit te schakelen.

Van al deze mogelijkheden vindt U in het onderstaande tenminste een toepassing. Ik spreek verder af, dat ik adressen hexadecimaal aanduid als offset t.o.v. het startadres van het in het geheugen geladen programma CHESSC.

Door de wijze van laden was dat startadres in mijn geval steeds hex. 50000. En de vervanging doe ik met een heel simpel programma in Basic, met behulp van poke- en peekinstructies. Dat programma geef ik aan het slot van dit verhaal, onder de naam CHAPRO.

Op 480E e.v. staat 4EBAFD70, zijnde JSR 4580. En die subroutine leidt, na wat voorbereidende instructies, naar de vectored utility 12A, of wel MD.SECTR : read a sectorheader. Het programma duikt dus in QDOS; het komt weer te voorschijn in de terugkeeradressen 3912, 3914 en 3916.

Hier gaat het programma verder bij respectievelijk "bad medium", "bad sector header", en "OK". Verder merken we op, dat de leesbuffer, waarin de sectorheader wordt geplaatst, begint op 3552, een adres waarheen wordt verwezen door de inhoud van het adres waarheen register A1 wijst. Pikant is dat met deze leesbuffer een stukje programma wordt overschreven, dat op dat moment niet meer nodig is: men kan dus nooit (ook niet tijdens het exploratiewerk) het programma herstarten na de leessequence gehad te hebben, tenzij men reset en herlaadt.

In de leesbuffer, die 14 bytes lang is, komt op de 13de en 14de plaats, dus op de adressen 355E en 355F, het z.g. randomnummer van de gelezen cartridge te staan, een soort betrekkelijk onzichtbaar kenteken daarvan. Op de plaatsen 3, t.e.m. 12, staat de naam van de gelezen cartridge. Het blijkt nu, dat het programma verderop het gelezen randomnummer volgens een bepaald recept (negate, AND met 03, ROR 3 plaatsen) omvormt en dan nagaat (op 4A42), of het resultaat hetzelfde is als een waarde, die het programma zelf meedraagt op de adressen 1D22 en 1D23. Lijkt deze beschrijving al behoorlijk slinks, het is niet het complete rookgordijn. Want eerst (op 4A3E) wordt een nietszeggend adres (4B35) geladen in A2. In de CMP instructie op 4A42 wordt dan met offset -2E13 het echte adres 1D22 berekend waar het controlegetal in het programma staat. Dat wordt dan vergeleken met (moet gelijk zijn aan) de inhoud van D2. In dat register is even eerder de inhoud van A28C overgeschreven, en die inhoud is berekend (als boven omschreven) uit de twee randombytes van de leesbuffer. Kan het kronkeliger? Ik geef dit hier zo uitvoerig weer om duidelijk te maken op welke wijze de softwareschrijvers hun werk ondoorzichtig trachten te maken. Daartoe behoort ook, dat dit hele inleidende gebeuren met wilde sprongen kunstmatig is verdeeld over een

gebied tot 20 Kbytes na het program-mabegin.

Om dit gehele doolhof te omzeilen gaan we op 4A2C niet de inhoud van A28C maar van 1D22 overschrijven in register D2. In A3 staat dan 8600 en met offset 9722 wijst dat naar 1D22.

Dus:

Wijziging 1

Op 4A2C (.L) staat 342B1C8C
Dit wordt 342B9722.

Zoals gezegd, willen we de hele leesoperatie overslaan. We willen daartoe springen van 480E naar 3922, maar omdat we nog wat anders moeten invoegen springen we naar 391A. De 8 bytes van 391A t.e.m. 3921 vullen we bij wijziging 3 weer op.

Nu dus:

Wijziging 2

Op 480E (.L) staat 4EBAFD70
Dit wordt 6000F10A.

De tweede test vinden we op 3650. Daar staat: 3650 B27AFF02
CMP.W \$00003554(PC),D1
Op 3554 (.W, dus met 3555) staan in de leesbuffer de eerste twee letters van de gelezen naam. Als het goed ging waren dat de C en de H, met als ASCII codes 6368. En vlak tevoren, op 3648, wordt #\$6368 geladen in D1.

We laten dit gehele mechanisme nu eens intact (over efficiëntie praten we even niet, maar U zult het verschil niet merken) en gaan de vereiste getallen 63 en 68 gewoon schrijven op de betreffende plaatsen van de overigens ongebruikte leesbuffer. Voor de hiertoe dienende instructies gebruiken we de bij wijziging 2 vrijgehouden 8 bytes.

Dus:

Wijziging 3

(a) Op 391A (.L) staat 548F205F.
Dit wordt 4BFAFC38 LEA #\$3554,A5
(b) Op 391E (.L) staat 70024E42.
Dit wordt 3ABC6368, ofwel

MOVE.W #\$6368,(A5)

Tenslotte moet nog wat overbodig en

dus hinderlijk geworden schermtekst en een aanslag van <SP> worden weggewerkt. We doen dat door van 4800 naar 480A te springen, door de instructies op 1F72 en 1F7C uit te vegen (d.w.z. vervangen door 4E714E71,NOPNOP) en door van 1F86 naar 23C6 te springen.

Dus:

Wijziging 4

(a) Op 4800 (.W) staat 4EBA
Dit wordt 6008 BRA.S 480A
(b) Op 1F72 (.L) staat 61000418
Dit wordt 4E714E71 NOPNOP
(c) Op 1F7C (.L) staat 6100040E
Dit wordt 4E714E71 NOPNOP
(d) Op 1F86 (.L) staat 60000404
Dit wordt 6000043E BRA 23C6

NASCHRIFT

Ik nodig U uit, om de hier aangegeven wijzigingen aan te brengen met behulp van mijn programma CHAPRO.U begint daartoe met Irun xxxx_CHAPRO en daarbinnen volgt U de schermaanwijzingen. Het te bewerken programma is CHESSC van de originele cartridge, met een lengte van 64K of te wel 65536 bytes. Save het resultaat onder een andere naam op een andere cartridge, en pas de bootfiles zo nodig aan. De omzetting naar disk gebeurt eveneens in de bootfiles, dus in Basic, en levert geen problemen.

U vindt het programma CHAPRO of als listing hierbij of in de biblio-

theek, dat moet de redactie uitmaken. Ik heb ook een gecompileerde versie SCCHAPRO_EXE aangeboden voor plaatsing in de bibliotheek.

Verder ga ik nog trachten, een BREAK zonder RESET te realiseren in CHESSC. Dat dat niet mee zal vallen is me al wel duidelijk geworden; het duurt dus vast nog wel even voordat ik daarover zal kunnen berichten. En voor degenen die mij met vragen, mededelingen of ervaringen willen verblijden: mijn adres is Pansierstraat 44, 2584EK Scheveningen; telefoon 070-556586.

Veel genoeg!

Pim van Leening

```

100 REMark *****
110 REMark ***** program CHAPRO *****
120 REMark ***** voor het aanbrengen van veranderingen in m.c. programma's *****
130 REMark ***** Copyright 1988 W.P.van Leening Scheveningen Holland *****
140 REMark *****
150 DIM devfi$(16),ra$(4),sa$(8),keus$(8),ad$(8),nc$(8),in$(1)
160 DIM lpd$(8),desfi$(16),yn$(1),x$(8)
170 MODE 4:WINDOW 512,256,0,0:CLS
180 OPEN #8,con_176x204a0x52:OPEN #7,con_336x204a177x52
190 OPEN #6,con_168x196a4x56
200 PAPER #6,2:PAPER #7,2:INK #6,7:INK #7,0:BORDER #8,4,0
210 AT 0,4:CSIZE 2,1:PRINT "PROGRAM CHAPRO ***** change MC program.":CSIZE 0,0
220 AT 2,6:CSIZE 0,1
230 INPUT "Which program to be changed (dev_file,ENTER if in memory) ? ";devfi$;
240 CSIZE 0,0
250 AT 4,6:PRINT "Adresses relative (r) or absolute (a) ? ";:INPUT ra$
260 sa$="00000":n=4:nb=0:np=2
270 CSIZE 0,0:AT #7,0,1:INPUT #7,"Length of program (bytes,dec) : ";lp
280 IF devfi$=""
290 AT #7,1,1:INPUT #7,"Starting address (HEX) : ";sa$
300 ELSE
310 base=RESPR(lp):sa$=hexa$(base,5):LBYTES devfi$,base
320 END IF
330 PAUSE 100:CLS #7
340 AT #6,0,2:PRINT #6,"Start addr. (HEX): ";sa$
350 AT #6,1,2:PRINT #6,"Length of program ";lp
360 AT #6,3,2:PRINT #6,"1. New address .B"
370 AT #6,4,2:PRINT #6,"2. New address .W"
380 AT #6,5,2:PRINT #6,"3. New address .L"
390 AT #6,6,2:PRINT #6,"4. Next address .B"
400 AT #6,7,2:PRINT #6,"5. Next address .W"
410 AT #6,8,2:PRINT #6,"6. Next address .L"
420 AT #6,10,2:PRINT #6,"7. Save results "
430 AT #6,11,2:PRINT #6,"8. QUIT,back to basic. "
440 AT #6,14,2:PRINT #6,"MAKE YOUR CHOICE! "
450 keus$=INKEY$(-1)
460 IF CODE(keus$)<48 OR CODE(keus$)>56:GO TO 450:ELSE :keus=keus$:CLS #6
470 IF keus=8:GO TO 720

```



```

480 IF keus=7:GO TO 650
490 vornb=nb:nb=keus:IF keus>3 THEN nb=nb-3
500 IF nb=3 THEN nb=4
510 AT #7,2,0:PRINT #7,"Te veranderen op (rel.) adres (HEX): ";
520 IF keus=4 OR keus=5 OR keus=6
530 IF n=4 THEN GO TO 340
540 ad=dec(ad$)+vornb:ad$=hexa$(ad,1):PRINT #7,ad$;" "
550 GO TO 580
560 END IF
570 INPUT #7,ad$;:PRINT #7," "
580 ad=dec(ad$):IF ra$=="R":ad=ad+dec(sa$)
590 AT #7,n,0:PRINT #7,hexa$(ad,5);" ";
600 FOR j=1 TO nb:PRINT #7,hexa$(PEEK(ad+j-1),2);
610 PRINT #7,FILL$(" ",12-2*nb);:nc$=""
620 FOR j=1 TO 2*nb:in$=INKEY$(-1):nc$=nc$&in$:PRINT #7,in$;
630 FOR j=1 TO nb:POKE ad+j-1,dec(nc$(2*j-1 TO 2*j))
640 n=n+1:GO TO 340
650 AT 4,0:CLS 2
660 AT 10,4:INPUT "Length of program (ENTER if unchanged) : ";lpd$
670 IF lpd$="" :lpd=lp:ELSE :lpd=lpd$
680 AT 12,4:INPUT "Destination file (device_file) : ";desfi$
690 SBYTES desfi$,dec(sa$),lpd:AT 4,0:CLS 2
700 AT 10,2:CSIZE 2,1:INPUT "Want another run (Y/N) ? ";yn$
710 IF yn$=="Y":CLS:GO TO 200
720 CLOSE#6:CLOSE #7:CLOSE #8:WINDOW 256,206,256,0
730 WINDOW #2,512,206,0,0:CLS #2:CLS:CLS #0:STOP
740 DEFine FuNction dec(x$)
750 dx=0
760 FOR i=1 TO LEN(x$)
770 da=x$(i) INSTR "0123456789ABCDEF"-1
780 dx=16*dx+da
790 END FOR i
800 RETURN dx
810 END DEFine dec
820 DEFine FuNction hexa$(ZZ,np)
830 LOCAL Z,xa$(8),q,r,r$(1)
840 xa$="":Z=ZZ
850 q=INT(Z/16):r=Z-16*q:Z=q
860 IF r<10 THEN r$=CHR$(48+r):ELSE :r$=CHR$(55+r)
870 xa$=r$&xa$
880 IF Z>0:GO TO 850
890 IF LEN(xa$)<np THEN xa$=FILL$("0",np-LEN(xa$))&xa$
900 RETURN xa$
910 END DEFine

```


Ombouw CST Disc Controller

Om dat CST een nieuwe Controller uitgebracht heeft met meerdere toolkit 2 comando 's en een RAM disk driver, en dit niet zondermeer werkt op een oude versie CST controller en het wel eens probleem oplevert met het aansturen van een 3 1/2" disc drives, heb ik een modificatie gemaakt op mijn controller om toch de nieuwe versie CST Eprom te kunnen gebruiken.

Voor de modificatie heeft men een 74LS133 (LS133) nodig een stukje wire wrapdraad een weerstand van 470 Ohm en de nieuwe versie CST Eprom.

Van de 74LS133 (LS133) buigt men alle pooten omhoog op pen 8 en pen 16 na. De LS133 soldeert men boven de 74LS266 (LS266) pen 8 van LS133 aan pen 7 van LS266 en pen 16 van LS133 aan pen 14 van LS266. (pennetjes iets naar elkaar

buigen).

Pennen 4 t/m 7 van LS 133 aan elkaar doorverbinden.

De pennen 1 t/m 4 en 10 t/m 15 van de LS 133 aan de adreslijnen A4 t/m A13 solderen (zie QL USER GUIDE voor de adreslijnen).

Het baantje dat aan onderkant van de controller naar p1 en p2 van LS03 loopt door snijden. Pen 9 van LS133 verbinden aan pen 1 en pen 2 van LS03.

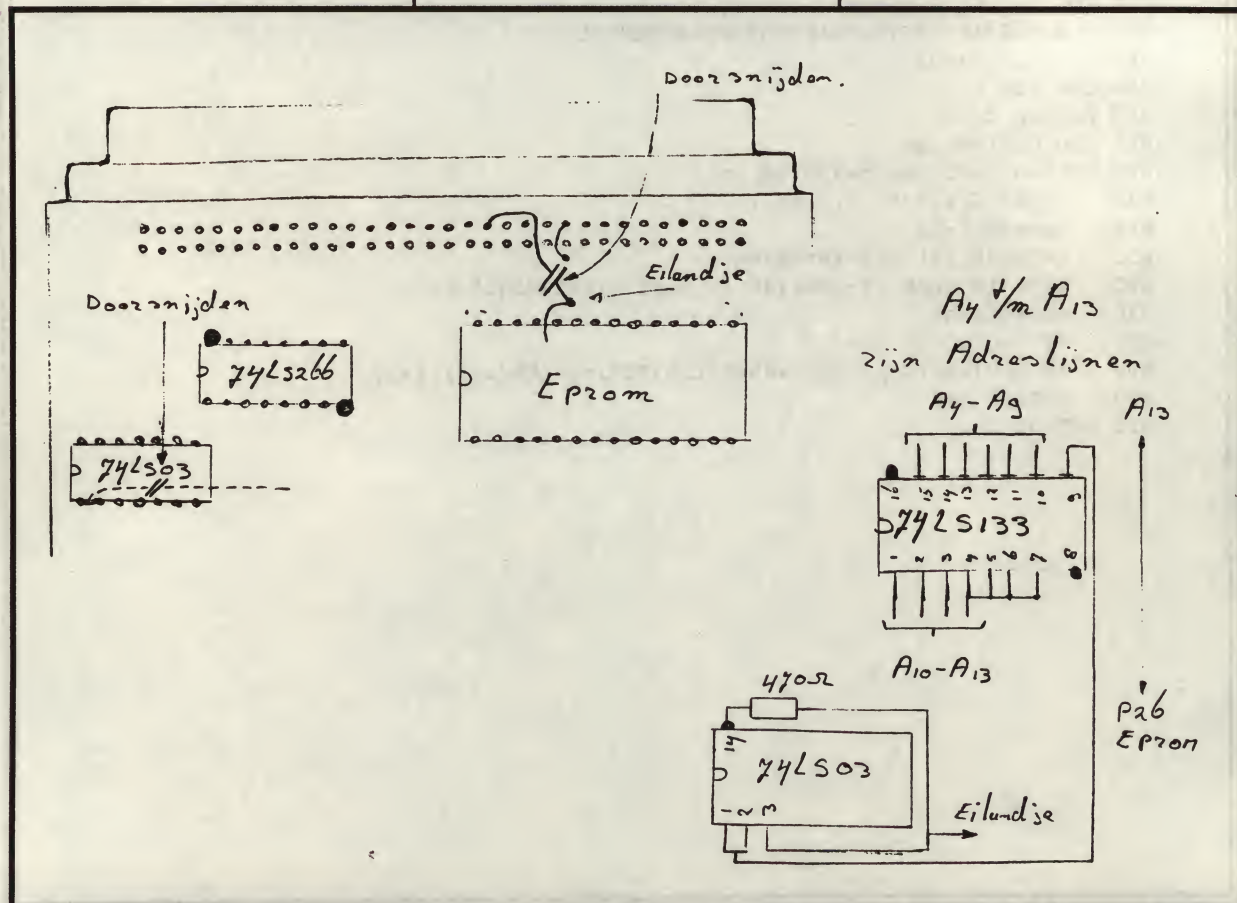
Nu moet er een baantje wat aan de bovenkant van de controller van adreslijn A13, a14 van de expantie poort, naar een eilandje loopt door snijden. Verbind het eilandje met pen 3 van de LS03. Soldeer de weerstand van pen 3 naar pen 14 van de LS03.

Aan de onder kant van de controller moet U nu een draadje leggen van pen 26 van de Eprom naar Adreslijn A13 solderen.

Als U nu alles goed gedaan heeft dan moet de controller weer werken.

Succes

Michel Spanjer



OPROEP

A.W.Boot.
DIJKSTR.105.
5554 PR VALKENSWAARD.
04902_15268.

VALKENSWAARD 23-06-88.

L.S.

WELKE QL-ER WERKTE ER MET JM-JS EN T.TEBBY SUPERQBOARD 512 EN MUIS?
OF MET QL EN QRAM.

WIE WIL VOOR BEGINNERS IN QUASAR UITLEGGEN HOE JE B.V. EEN PROGRAMMA
KUNT "BOOTMAKEen" ZODAT HET OPSTART MET RESET +F1.

WIE GEEFT EEN UITGEWERKT VOORBEELD HOE JE MET QRAM 4 PROGRAMMAS
KUNT MULTITASKEN.

WIE GEEFT EEN UITGEWERKT VOORBEELD HOE JE EEN PROGRAMMA KUNT HOT-
MAKEen ZODAT HET OPSTART MET AANKLIKKEN VAN DE MUIS .

WIE WEET WELKE PROGRAMMAS ONDER QRAM WILLEN WERKEN .MOGEN DIT
PROGRAMMAS ZIJN MET EEN BOOT ERVOOR?.WELKE VOORWAARDEN ZIJN ER?.

DE VOORBEELDEN MOETEN WEL ZEER UITGEBREID WORDEN VOORGEDAAN AND-
ERS SNAP IK HET NIET.

WIL DE REDACTIE MISSCHIEN HIEROVER ENKELE ARTIKELN SCHRIJVEN VOOR
ALLE BEGINNERS .DE QRAM BOEKJES HELPEN ME NIET GENOEG!.

WELKE QLer MET SUPERQBOARD 512 + MOUSE WIL MIJ EENS OP WEG
HELPEN ?.

VEEL HULP ZOU WELKOM ZIJN !.

BIJ VOORBAAT DANK,
JOS BOOT

PROCEDURE OM EEN MATRIX TE INVERTEREN/ VERMENIGVULDIGEN

Dit programma bevat een aantal procedures, waarvan de belangrijkste zijn: INV, PROD en DRUKAF. De rest van de procedures wordt aangeroepen door de procedure INV.

De procedure INV bepaalt de inverse van een matrix door hem "schoon te vegen".

De procedure wordt aangeroepen met: "INV matrix1,matrix2", waarbij matrix1 en matrix2 matrices met dezelfde dimensie moeten zijn. De procedure zet de inverse van matrix1 in matrix2. Als de dimensies van matrix1 en matrix2 niet gelijk zijn geeft het programma een melding in window 0, maar onderbreekt het programma niet. Er wordt dan natuurlijk geen inverse uitgerekent. De aanroep INV matrix1, matrix1 is dus ook korrekt en heeft als resultaat dat na de aanroep matrix1 de inverse van zichzelf is geworden.

De procedure PROD bepaalt het produkt van twee matrices, en wel matrix1 en matrix2 en zet het produkt in matrix3; hij wordt aangeroepen met: "PROD matrix1, matrix2, matrix3", waarbij ook weer de dimensies gecontroleerd worden.

Als bijv. matrix1 een NxM matrix is dan moet matrix2 een MxK matrix zijn en matrix 3 een NxK, waarbij n,m en k een willekeurige getallen zijn.

Dan tenslotte de procedure DRUKAF, deze wordt als volgt aangeroepen: "DRUKAF matrix", deze proce-

dure drukt dan de matrix af in kanaal #3, en de procedure werkt alleen fatsoenlijk bij kleine matrices als een rij op het scherm past.

Bovendien maakt deze procedure gebruik van het TOOLKIT II commando print_using en de procedure zal dus aangepast moeten worden indien U geen toolkit heeft. Aan het eind van het programma staan wat voorbeelden die U kunt bekijken door het programma te runnen. Het is natuurlijk ook mogelijk om deze procedures buiten een programma te gebruiken als een soort basic uitbreiding. Om uw programma te checken: de inverse van de inverse is gelijk aan het origineel en het produkt van een matrix met z'n inverse is gelijk aan de eenheidsmatrix.

Roelof Crevecoeur.

```

1000 REMark -----
1005 REMark Enkele procedures om matrices te inverteren en hun produkt
1010 REMark uit te rekenen.
1015 REMark Geprogrammeerd door: R. M. Crevecoeur
1020 REMark Laatste wijziging : 21 mei 1988
1025 REMark -----
1030 REMark Error procedure.
1035 REMark -----
1040 WHEN ERROR
1045     IF ERLIN THEN
1050         OPEN #8,scr_292x52a106x100
1055         BORDER #8,1,7
1060         CLS #8
1065         PRINT #8,"Error:Fatale error, programma wordt onderbroken."
1070         PRINT #8,"      Check de onafhankelijkheid van de matrix."
1075         PRINT #8,"      Fout is opgetreden in regnr: ";ERLIN
1080         PRINT #8,"      QDOS error nummer: ";ERNUM
1085         PRINT #8,"      QDOS omschrijving: ";:REPORT #8
1090         CLOSE #8
1095         STOP
1100     ELSE REPORT #0
1105     END IF
1110 END WHEN
1115 REMark -----
1120 REMark
1125 REMark
1130 REMark -----

```



```

1135 REMark Procedure die de inverse van "matrix1" uitrekent, en deze
1140 REMark weer mee terug geeft in "matrix2".
1145 REMark -----
1150 DEFine PROCedure INV(matrix1,inverse)
1155   LOCal matrix2(DIMN(matrix1,1),2*DIMN(matrix1,2)+1)
1160   IF (DIMN(matrix1,1)=DIMN(inverse,1) AND
      DIMN(matrix1,2)=DIMN(inverse,2)) THEN
1165     ReDefineMatrix2 matrix1,matrix2
1170     FOR teller2=0 TO DIMN(matrix2,1)
1175       MaakHoofdDiagonaal1 matrix2,teller2
1180       FOR teller1=0 TO DIMN(matrix2,1)
1185         MaakKolom0 matrix2,teller1,teller2
1190       END FOR teller1
1195     END FOR teller2
1200     ReDefineMatrix1 inverse,matrix2
1205   ELSE
1210     PRINT #0,"Error:De dimensies van de inverse zijn niet goed."
1215   END IF
1220 END DEFine
1225 REMark -----
1230 REMark
1235 REMark
1240 REMark -----
1245 REMark Deze procedure maakt het element in de kolom "kolom", dat op
1250 REMark de hoofddiagonaal ligt 1, door de rij te delen door het
1255 REMark element op de hoofddiagonaal.
1260 REMark -----
1265 DEFine PROCedure MaakHoofdDiagonaal1(matrix,rij)
1270   LOCal teller,n,factor
1275   LET n=0
1280   REPeat element_ongelijk_nul
1285     IF matrix(rij,rij+n)<>0 THEN
1290       EXIT element_ongelijk_nul
1295     ELSE
1300       LET n=n+1
1305     END IF
1310   END REPeat element_ongelijk_nul
1315   IF n<>0 THEN
1320     VeegRij matrix,rij,rij+n,1
1325   END IF
1330   LET factor=matrix(rij,rij)
1335   FOR teller=0 TO 2*DIMN(matrix,1)+1
1340     matrix(rij,teller)=matrix(rij,teller)/factor
1345   END FOR teller
1350 END DEFine DeelRij
1355 REMark -----
1360 REMark
1365 REMark
1370 REMark -----
1375 REMark Deze procedure maakt een element in de kolom "kolom" nul, als
1380 REMark dit niet op de hoofd diagonaal, ligt.
1385 REMark -----
1390 DEFine PROCedure MaakKolom0(matrix,rij,kolom)
1395   LOCal factor
1400   IF rij<>kolom THEN
1405     LET factor=matrix(rij,kolom)
1410     VeegRij matrix,rij,kolom,factor
1415   END IF
1420 END DEFine MaakKolom0
1425 REMark -----

```



```

1430 REMark
1435 REMark
1440 REMark -----
1445 REMark Deze procedure trekt "factor"*rij2 van rij1 af in "matrix".
1450 REMark -----
1455 DEFine PROCedure VeegRij(matrix,rij1,rij2,factor)
1460     LOCAL teller
1465     FOR teller=0 TO 2*DIMN(matrix,1)+1
1470         LET matrix(rij1,teller)=matrix(rij1,teller)-
            matrix(rij2,teller)*factor
1475     END FOR teller
1480 END DEFine VeegRij
1485 REMark -----
1490 REMark
1495 REMark
1500 REMark -----
1505 REMark Deze procedure plakt de eenheids matrix aan "matrix" en zet de
1510 REMark waarden vervolgen in "array".
1515 REMark -----
1520 DEFine PROCedure ReDefineMatrix2(mat1,mat2)
1525     LOCAL teller1,teller2
1530     FOR teller1=0 TO DIMN(mat2,1)
1535         FOR teller2=0 TO DIMN(mat2,1)
1540             LET mat2(teller1,teller2)=mat1(teller1,teller2)
1545             IF teller1=teller2 THEN
1550                 LET mat2(teller1,teller2+DIMN(mat2,1)+1)=1
1555             END IF
1560         END FOR teller2
1565     END FOR teller1
1570 END DEFine RedefineMatrix
1575 REMark -----
1580 REMark
1585 REMark
1590 REMark -----
1595 REMark Deze procedure zet de iverse matrix ("mat2") in "mat1".
1600 REMark -----
1605 DEFine PROCedure ReDefineMatrix1(mat1,mat2)
1610     LOCAL i,j
1615     FOR i=0 TO DIMN(mat1,1)
1620         FOR j=0 TO DIMN(mat1,2)
1625             mat1(i,j)=mat2(i,j+DIMN(mat1,1)+1)
1630         END FOR j
1635     END FOR i
1640 END DEFine ReDefineMatrix1
1645 REMark -----
1650 REMark
1655 REMark
1660 REMark -----
1665 REMark Deze procedure drukt de array "matrix" af.
1670 REMark -----
1675 DEFine PROCedure DRUKAF(matrix)
1680     PRINT #3
1685     FOR a=0 TO DIMN(matrix,1)
1690         FOR b=0 TO DIMN(matrix,2)
1695             PRINT USING #3,"-####.###",matrix(a,b)!
1700             REMark Dit kommando vervangen als U geen TOOLKIT II heeft
1705         END FOR b
1710         PRINT #3
1715     END FOR a
1720     PRINT #3

```



```

1725 END DEFine DRUKAF
1730 REMark -----
1735 REMark
1740 REMark
1745 REMark -----
1750 REMark Deze procedure bepaalt het product van 2 matrices
1755 REMark -----
1760     DEFine PROCedure PROD(matrix1,matrix2,prodmatrix)
1765     IF NOT(DIMN(matrix1,2)=DIMN(matrix2,1)) THEN
1770         PRINT #0,"Error:Geen product mogelijk."
1775     ELSE
1780         IF NOT(DIMN(matrix1,1)=DIMN(prodmatrix,1) AND
1785             DIMN(matrix2,2)=DIMN(prodmatrix,2)) THEN
1790             PRINT #0,"Error:Product matrix onjuist gedefineerd."
1795         ELSE
1800             FOR kolom=0 TO DIMN(prodmatrix,1)
1805                 FOR rij=0 TO DIMN(prodmatrix,2)
1810                     prodmatrix(kolom,rij)=0
1815                 END FOR rij
1820             END FOR kolom
1825             FOR kolom=0 TO DIMN(prodmatrix,1)
1830                 FOR rij=0 TO DIMN(prodmatrix,2)
1835                     FOR teller=0 TO DIMN(matrix1,2)
1840                         prodmatrix(kolom,rij)=prodmatrix(kolom,rij)+
1845                         matrix1(kolom,teller)*matrix2(teller,rij)
1850                     END FOR teller
1855                 END FOR rij
1860             END FOR kolom
1865         END IF
1870     END IF
1875 END DEFine PROD
1880 REMark -----
1885 REMark -----
1890 DIM voorbeeld1(3,3)           :REMark Declareer enkele voorbeeld matrices
1895 DIM voorbeeld2(3,3)
1900 DIM voorbeeld3(3,3)
1905 DIM voorbeeld4(4,6)
1910 DIM voorbeeld5(6,8)
1915 DIM voorbeeld6(4,8)
1920 REMark
1925 REMark
1930 RESTORE 2225
1935                                     REMark Lees matrix "voorbeeld1" in
1940 FOR a=0 TO DIMN(voorbeeld1,1)
1945     FOR b=0 TO DIMN(voorbeeld1,2)
1950         READ voorbeeld1(a,b)
1955     END FOR b
1960 END FOR a
1965                                     REMark Lees matrix "voorbeeld4" in
1970 FOR a=0 TO DIMN(voorbeeld4,1)
1975     FOR b=0 TO DIMN(voorbeeld4,2)
1980         READ voorbeeld4(a,b)
1985     END FOR b
1990 END FOR a
1995                                     REMark Lees matrix "voorbeeld5" in
2000 FOR a=0 TO DIMN(voorbeeld5,1)
2005     FOR b=0 TO DIMN(voorbeeld5,2)
2010         READ voorbeeld5(a,b)

```



```

2015     END FOR b
2020 END FOR a
2025 REMark -----
2030 REMark
2035 REMark
2040 REMark -----
2045 OPEN #3,scr_512x216a0x0:CLS #3 :REMark Open uitvoer device (scherm)
2050 REMark OPEN #3,ser1           :REMark Open uitvoer device (printer)
2055 REMark -----
2060 PRINT #3,"Matrix 'voorbeeld1'"
2065 DRUKAF voorbeeld1
2070 INV voorbeeld1,voorbeeld2
2075 PRINT #3
2080 PRINT #3,"Matrix 'voorbeeld2', dit is de inverse van 'voorbeeld1'"
2085 DRUKAF voorbeeld2
2090 PROD voorbeeld1,voorbeeld2,voorbeeld3
2095 PRINT #3
2100 PRINT #3,"Matrix 'voorbeeld3', dit is het product van 'voorbeeld1'"
2105 PRINT #3,"en de inverse hiervan die in 'voorbeeld2' staat"
2110 DRUKAF voorbeeld3
2115 INV voorbeeld2,voorbeeld3
2120 PRINT #3
2125 PRINT #3,"Matrix 'voorbeeld3', die de inverse van 'voorbeeld2' is, "
2130 PRINT #3,"maar omdat 'voorbeeld2' de inverse van 'voorbeeld1' is, "
2135 PRINT #3,"is dit de inverse van de inverse van 'voorbeeld1' en dus "
2140 PRINT #3,"gelijk aan 'voorbeeld1' zelf."
2145 DRUKAF voorbeeld3
2150 PRINT #3
2155 PRINT #3,"Matrix 'voorbeeld4'"
2160 DRUKAF voorbeeld4
2165 PRINT #3
2170 PRINT #3,"Matrix 'voorbeeld5'"
2175 DRUKAF voorbeeld5
2180 PROD voorbeeld4,voorbeeld5,voorbeeld6
2185 PRINT #3
2190 PRINT #3,"Matrix 'voorbeeld6' met het produkt van 'voorbeeld4' en"
2195 PRINT #3,"'voorbeeld5'"
2200 DRUKAF voorbeeld6
2205 REMark -----
2210 CLOSE #3           :REMark Sluit het uitvoer device
2215 STOP
2220 REMark -----
2225 DATA 10,-2,2,-1    :REMark data van "voorbeeld1"
2230 DATA -2,42,-12,0
2235 DATA 2,-12,118,-8
2240 DATA -1,0,-8,9
2245 REMark -----
2250 DATA 1,2,3,4,5     :REMark data van "voorbeeld4"
2255 DATA 34,23,1,.4,.5
2260 DATA 2,.4,1,2E-2,3
2265 DATA 1E-3,.3,.34,3E-3,4
2270 DATA 1,4,3,45,2
2275 DATA 2,34,1,23,56
2280 DATA 0,45,3,2,1
2285 REMark -----
2290 DATA 1,2,3,4,5,6,7 :REMark data van "voorbeeld5"
2295 DATA .3,3E-2,3E-3,.2,.8,.9,5
2300 DATA 2,3,5,3,23,12,1
2305 DATA 2,45,3,23,12,5,67
2310 DATA 23,4,1,5,3,4,2

```


2315 DATA 2,5,3,7,4,5,23
 2320 DATA 1,45,3,2,67,2,3
 2325 DATA 8,5,3,4,12,45,8
 2330 DATA 4,3,56,4,21,1,2
 2335 REMark -----

Matrix 'voorbeeld1'

10.000	-2.000	2.000	-1.000
-2.000	42.000	-12.000	0.000
2.000	-12.000	118.000	-8.000
-1.000	0.000	-8.000	9.000

Matrix 'voorbeeld3', die de inverse van 'voorbeeld2' is, maar omdat 'voorbeeld2' de inverse van 'voorbeeld1' is, is dit de inverse van de inverse van 'voorbeeld1' en dus gelijk aan 'voorbeeld1' zelf.

10.000	-2.000	2.000	-1.000
-2.000	42.000	-12.000	0.000
2.000	-12.000	118.000	-8.000
-1.000	0.000	-8.000	9.000

Matrix 'voorbeeld2', dit is de inverse van 'voorbeeld1'

0.102	0.005	-0.001	0.011
0.005	0.025	0.003	0.003
-0.001	0.003	0.009	0.008
0.011	0.003	0.008	0.120

Matrix 'voorbeeld4'

1.000	2.000	3.000	4.000	5.000	34.000	23.000
1.000	0.400	0.500	2.000	0.400	1.000	0.020
3.000	0.001	0.300	0.340	0.003	4.000	1.000
4.000	3.000	45.000	2.000	2.000	34.000	1.000
23.000	56.000	0.000	45.000	3.000	2.000	1.000

Matrix 'voorbeeld3', dit is het product van 'voorbeeld1' en de inverse hiervan die in 'voorbeeld2' staat

1.000	0.000	0.000	0.000
0.000	1.000	0.000	0.000
0.000	0.000	1.000	0.000
0.000	0.000	0.000	1.000

Matrix 'voorbeeld5'

1.000	2.000	3.000	4.000	5.000	6.000	7.000	0.300	0.030
0.003	0.200	0.800	0.900	5.000	2.000	3.000	5.000	3.000
23.000	12.000	1.000	2.000	45.000	3.000	23.000	12.000	5.000
67.000	23.000	4.000	1.000	5.000	3.000	4.000	2.000	2.000
5.000	3.000	7.000	4.000	5.000	23.000	1.000	45.000	3.000
2.000	67.000	2.000	3.000	8.000	5.000	3.000	4.000	12.000
45.000	8.000	4.000	3.000	56.000	4.000	21.000	1.000	2.000

Matrix 'voorbeeld6' met het produkt van 'voorbeeld4' en 'voorbeeld5'

1466.006	2607.400	218.600	206.800	1755.000	408.000	688.000	438.300	498.030
151.401	122.440	16.700	12.020	50.620	28.580	31.520	34.320	20.970
85.695	293.429	22.682	27.953	118.220	43.991	62.266	22.320	52.282
1296.009	2886.600	153.400	223.700	2408.000	391.000	1205.000	787.200	654.120
3102.168	1243.200	322.800	208.400	707.000	468.000	539.000	520.900	293.690

Wie heeft er ervaring met het gebruik van
 Harddisk interface van CST

Reacties gaarne naar Ard Jonker Tel 020 - 230795

CURSUS MACHINETAAL Hoofdstuk 9

SUBROUTINES

Dit zijn routines die vaker gebruikt worden. Ze worden geschreven als stukjes programma, op schijf of cartridge opgeslagen en wanneer nodig in een programma gebruikt.

Meestal zijn dit routines die een algemene toepasbaarheid hebben.

Voorbeelden hiervan zijn sorteer algoritmes, speciale input en output routines etcetera. Een subroutine moet vaak een zogenaamd argument mee krijgen. Stel dat we een subroutine hebben gemaakt die een letter op het scherm tovert. We kunnen dan wel de routine aanroepen, maar hoe weet deze routine dan welke letter hij (zij) moet afdrukken? Dit kunnen we doorgeven via de registers. Aangezien we maar een beperkt aantal registers hebben kunnen dit niet te veel parameters zijn. Een routine die een letter op het scherm zet, en het gebruik van deze routine kunt u zien in het onderstaande programma. Het gaat om een routine die steeds karakters afdruckt tot de gebruiker een 'Q' intoeft.

main_program

```
BSR get_a_key
```

* kijk welke toets gebruiker heeft ingetoeft.

* na terugkeer staat de ascii-code van de toets in D0

```
BSR print_key
```

* druk een karakter af. De ascii waarde van het karakter in D0

```
CMP 'Q',D0
```

* wil de gebruiker stoppen? zo niet, spring dan weer terug.

```
BNE main_program
STOP
```

* anders: stop de processor.

* deze subroutine drukt een ascii karakter af op het beeldscherm. In D0 moet de waarde van het karakter staan

```
print_key
BSR druk_af
```

* het eigenlijke afdrukken gebeurt in een speciale routine.

* Die kijkt ook waar op het scherm het karakter moet verschijnen. Is dit gebeurt, keer dan terug naar het programma dat je aanriep.

```
RTS
```

* deze subroutine scant het keyboard. Als er een toets is ingedrukt geeft hij de ascii-waarde van deze toets terug in D0.

```
get_a_key
BSR scan_keyboard
```

* kijk of er een toets is ingedrukt. Als er geen toets is ingedrukt dan zal de routine scan_keyboard het getal 0 teruggeven

```
CMPI.B #0,D0
BEQ get_a_key
```

* als er geen toets is ingedrukt, scan dan opnieuw het toetsenbord

```
RTS
END einde van het programma
```

Allereerst een paar nieuwe mnemonics: STOPen RTS. STOP is de instructie om de processor te laten stoppen. Dit is een extreem commando, de enige bruikbare manier om nog verder te gaan is een RESET (tenzij U speciale voorzorgsmaatregelen neemt, maar dat is alleen voor de ver gevorderde gebruiker). De instructie RTS (ReTurn from Subroutine) zorgt ervoor dat het programma weer terugkeert naar de plaats waar de subroutine aanroep (BSR of JSR) plaats vond. De daarop volgende instructie zal nu worden uitgevoerd.

Het mechanisme dat hierachter steekt is eenvoudig, maar o zo doordacht! Als de processor de instructie BSR tegenkomt, zet hij het adres van de instructie die direct volgt op BSR op de stack. De 'stack' is een stuk geheugen. Het wordt aangewezen door de StackPointer, A7. Dit register heeft U al eerder gezien bij de processor beschrijving. Een getal wordt 'op stack' gezet door de inhoud van A7 met het juiste aantal bytes te verlagen, waarna het te verplaatsen getal naar de geheugenplaatsen waar A7 naar wijst wordt verzet. Sla er even een boek met plaatjes op na als U dit niet direct kan volgen.

Een adres is altijd vier bytes lang (32 bits). A7 wordt dus eerst bijgewerkt zodat hij naar een volgende locatie op de stack wijst. Dit bijwerken houdt in dat de inhoud van A7 met 4 wordt verlaagd voor een Long word (een adres is altijd een longword,

maar niets verhindert U om ook gegevens met andere afmetingen op stack te zetten).

Op deze wijze kunnen we steeds weer getallen op stack zetten. De stack groeit omlaag. Dat wil zeggen dat hoe meer getallen er op stack staan, hoe lager A7 wijst. U kunt zelf ook dingen op stack zetten (alhoewel dit dodelijk is voor uw programma's als U vergissingen begaat) met de instructie

```
MOVE.L    ...,-(A7)
```

Dit betekent dat de stackpointer verlaagd word (daarom staat de '-' ook voor de instructie), en de 4 bytes '...' op de geheugenplaatsen (A7) tot 4 (A7) komen te staan. '(A7)' betekent waar A7 naar wijst' en '4(A7)' betekent 'vier geheugen plaatsen hoger dan waar A7 naar wijst'. Bevat A7 bijvoorbeeld #100, dan betekent

```
MOVE.L    D0,-(A7)
```

dat de vier bytes lange inhoud van D0 naar de geheugenplaat-

sen 97,98,99 en 100 geschreven moet worden, waarna A7 '96' bevat. Het terughalen van dit long word naar D0 gaat door

```
MOVE.L    (A7)+,D0
```

oftewel: haal de top van stack af, en zet die in D0.

Verhoog daarna A7 weer met 4.

Op het gebruik van de stack komen we nog uitgebreid terug als we het in een andere context gaan hebben over het doorgeven van gegevens tussen subroutines.

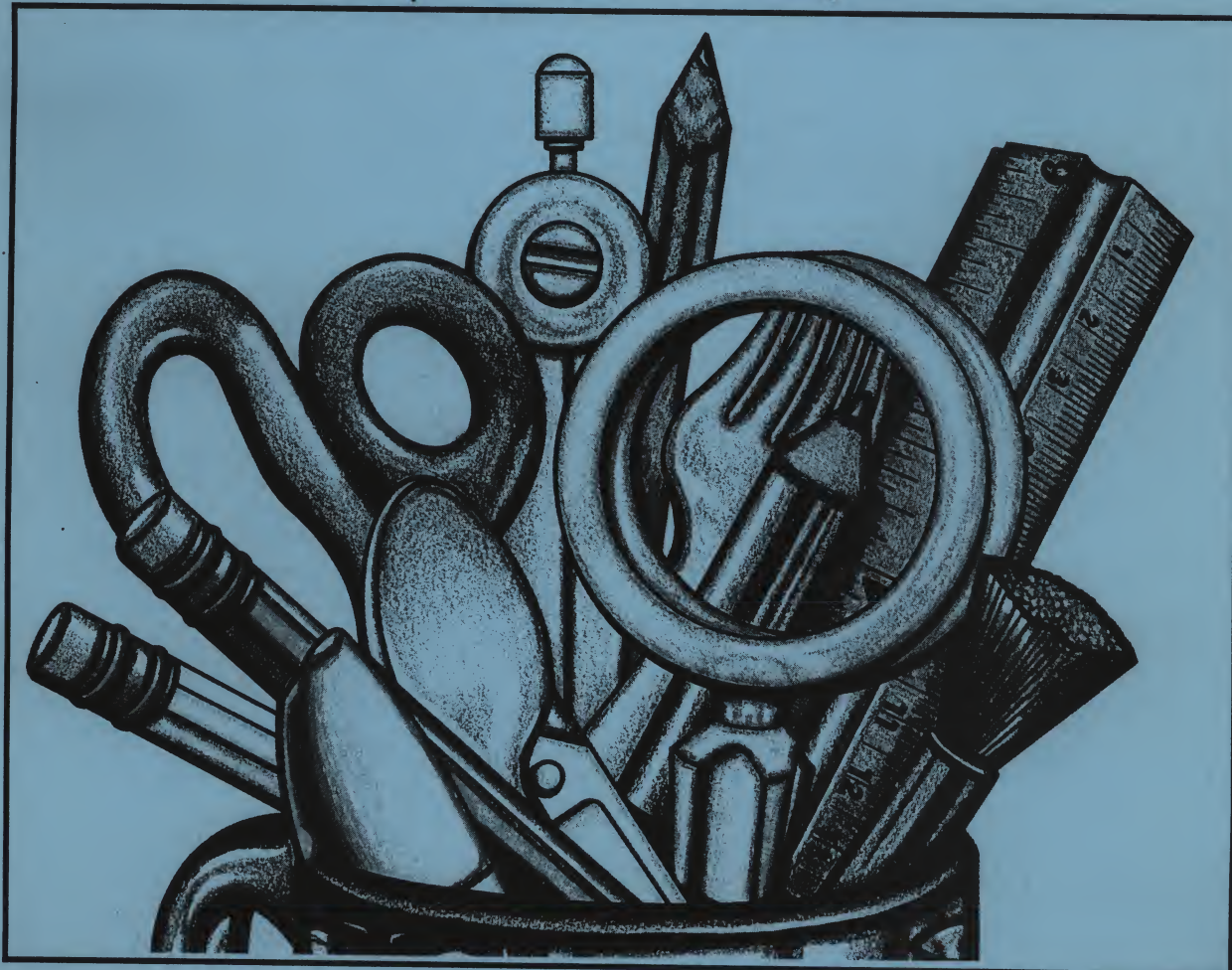
Terug naar ons voorbeeld. U ziet dat we gebruik maken van veel commentaar. Dit is als het zinnig commentaar is, zeer belangrijk. Omdat machinetaal toch moeilijk te doorgronden is, zeker als U de wat geavanceerder truiks onder de knie begint te krijgen en op slinkse wijze snelheidswinst probeert te krijgen, of met gemene truiks de laatste bits uit uw programma probeert te werken om het zo compact mogelijk te laten zijn. Verder gebruik ik zelfs voor de meest simpele dingen een subroutine. Dit heeft tot gevolg dat later, als er

iets verandert aan bijvoorbeeld het toetsenbord, alleen deze routine hoeft te worden aangepast. Aan het hoofdprogramma hoef ik niets te veranderen. De subroutine levert - hoe hij dan ook moge werken- altijd de ingedrukte toets als ascii waarde in D0 af.

Hier komt ook een ander voordeel naar boven. Doordat er precies is gedefinieerd hoe een bepaalde routine zijn gegevens aangeleverd wil krijgen, en hoe hij zijn resultaten terug geeft, hoeven we niets te weten van het interne van de routine.

Hij werkt, en daarmee weten wij genoeg. Het enige wat ik voor de afdruk-routine hoef te weten, is dat hij zijn karakter in D0 wil hebben aangeleverd. Hoe de routine weet waar hij het karakter op het scherm zetten moet, hoef ik niet te weten. Het wordt afgedrukt, en daarmee uit.

Ard Jonker



Vraag en Aanbod

Te koop aangeboden:

QL JS.....f.225,-
QEP II Epromprogrammer

Fred v.d. Neut
01807-10553

Te koop aangeboden:

Ferguson Monochrome monitor,
kleur amber, inclusief QL verbin-
dingskabel, nog geen jaar oud
Prijs nader overeen te komen.

T. Blom
079 - 515427

Te koop aangeboden:

QL JS 384 KB
Compleet met Sandy Super Q
Board, Monitor monochrome,
real time clock, TEAC disk-drive
met voeding dit alles ingebouwd
in een IBM-kast, met veel
software op diskettes compleet
met manuals.....f..900,--

Wil Langezaal
Tel 03402 - 64026

Te koop aangeboden:

IBM toetsenborden NIEUW
101 toetsen.....f..249,-

PC XT kasten NIEUW
voor inbouw van de QL....f 159,-

Klaas Zeven
Tel. 05910 - 24087
Alleen ma.di.do middag
van 14.00 - 17.00 uur.

Te koop gevraagd:

Geheugenuitbreiding 512 K,
Trumpcard o.i.d.

Laurens
Tel. 05495 - 1864
Na 18.00 uur

Te koop aangeboden:

QL JM 128 K, CUB 653 kleuren-
monitor, 3.5 inch Sanyo dubbel-
drive (720 KB), M.P. diskinterface
met geradts ROM, Printer Brother
HR5 (RS 232), Miracle Modem,
Miracle Centronics interface,
TTTtoolkit (ROM), 45 cartridges,
11 boeken, veel software o.a.
PSION CHESS en ADDERSOFT
Assembler.
Totaal.....f..1.500,--

Tel. 070 - 270376
(na 18.00 uur)

Te koop aangeboden

QL-JS versie + 640 K geheugen-
uitbreiding + Diskinterface + 2
disk drives een 5 1/4 en een 3 1/2
inch + erg veel software +erg veel
boekwerken + toolkit II en ICE op
eprom + 50 cartridges alle in
100% staat prijs.....f..1250,--

P.M.P. Moonen
Tel. 04166 - 2617

ANTWOORD

Op de vraag wie de auteur was
van het artikel over het PC-
toetsenbord is het antwoord

Frank Troost
Tel. 01672 - 2584

Te Koop aangeboden:

QL512(JS) met monitor(Philps
BM 7552 g/w, 25 cartridges met
software, en 2 eproms met houd-
er; prijs f 600,--

Eirk Hoogcarspel
Tel 010-415 70 97

Te koop aangeboden:

Seikosha SP 1000
Serieel f 300,--

A.G. Vermeulen
Goudenregenlaan 12
Castricum
Tel. 02518 - 58723

ANTWOORD

Als reactie op de vraag van Ed
Kats (Quasar 35,blz 721) inzake
sorteren in Archive nog het vol-
gende.

Zijn probleem met een sleutelveld
van 10 cijfers is geen probleem,
wanneer je die 10 cijfers op de
een of andere manier in twee vel-
den onder brengt.

Archive kent namelijk het gekop-
peld sorteren op twee (maximaal
zelfs vier) velden van ten hoogste
8 cijfers elk. Je kunt dus tot 32 cij-
fers terecht.

De opdracht wordt dan b.v. OR-
DER num1;a,num2;a (voor op-
klimmende rangschikking naar de
inhoud van veld num1, gekoppeld
aan veld num2).

W.P. van Leening
Pansierstraat 44
2584EK Scheveningen